GETS32 States Support

Implementation plan

Mon, Sep 10, 2007

For the `GETS32` protocol, data requests can specify a Data Event String that prescribes when replies are to be returned. The four formats of `DES` are called Immediate (one-shot), Periodic, Event, and State. This note describes the support for State-based data requests.

What are States?

As used in the `GETS32` protocol, States are similar to clock events, but they are not present in hardware; instead, they are delivered only via a multicast message to which all nodes that support State-based requests listen. Each State has an Acnet device name of the form `V:xxxxxx`. The States multicast message delivers one or more State records that specify device index values (DI's) rather than names to make it easier for front ends to process. The format of a State record is:

Field	*Size*	*Meaning*
`xtra`	2	?
`devInx`	4	Acnet device index, as 140219, for `V:FEDN`
`devVal`	2	16-bit State device value
`gpsSec`	4	GPS time-of-day, seconds since 1970
`gpsNano`	4	GPS time-of-day, nanoseconds of current second

Each State record occupies 16 bytes. From inspection, each multicast message nearly always includes only a single State record, with occasional examples of two records. Rarely, there occur examples of more than 2, even as many as 30 records. Perhaps 64 is a useful maximum.

A distinction with States as compared with clock events is the 16-bit value that is delivered along with the State device DI and the time stamp. A data request can thus ask for replies on a State when its value matches a specified reference value.

Here is the format of a State-based request Data Event String:

```
"s,V:CLDRST,9,1000,="
```

(In this example from the `GETS32` documentation, a State device name is used. But for the support described herein, it will be assumed that a device index is used.) This example asks for replies when the given State occurs, and the State value = 9, with a delay of 1000 ms. Besides "`=`", the other operators allowed are "`!=`" for not equal, and "`*`" for any; *i.e.*, the State value is ignored.

Note that successive announcements of a given State can indicate the same value. A reply is to be sent for any announcement where the value qualifies with the given operator and reference value, no matter whether it has changed since the last announcement, or not.

Implementation

As part of the implementation, a State Pool Table (`SPT`) is built and updated by a local application called `STTE` that processes the multicast messages. Its has 16-byte elements:

Field	*Size*	*Meaning*
`devX`	4	Acnet device index
`val`	2	16-bit State device value
`cnt`	2	Count of occurrences
`gmtSec`	4	GMT time-of-day, seconds since 1900
`gmtUs`	4	GMT time-of-day, microseconds of current second

Note that the time stamp format is modified from that in the multicast message to better match that maintained by the front end system code, originally developed for MiniBooNE.

This `SPT` holds the most recent State data as seen from the multicast message. In order to detect a new occurrence of a given State, watch for a change in the `cnt` field, not the `val` field.

GETS32 handling

The `GETS32` (and `RETDAT`) support uses a common structure for the parameters that are specified in the Data Event String (or the `RETDAT FTD`). The structure is packed into two 32-bit long words. The details of the structure depend on the type of `DES` used. For the State-based case described here, the fields used are as follows:

Field	*#Bits*	*Meaning*
`flags`	4	`GETS32` flag in ms bit
`type`	4	`DES` type#, where 0=immediate, 1=periodic, 2=event, 3=state
`op`	4	operator code, where 0=equal, 1=not equal, 2=any
`delay`	20	delay after State in ms, range 0–17 minutes
`valu`	16	reference State value
`indx`	12	index to `SPT` entry, range 1–4095
`cntr`	4	counter bits from `SPT` entry, range 0–15

Details

The `ScanDES` function parses the State-type string. It searches the `SPT` to find a match with the specified DI. If it fails to find a match, it creates a new `SPT` entry for that DI, initializing the `cnt` field to zero. The various specified fields are packed into the above structure.

The `ReplyDue` function performs the logic check to determine, for a given active request, whether it is time for a reply to be built; *i.e.*, it determines whether a reply is due this cycle. For a State-based request, check whether the State has occurred by comparing the low 4 bits in the `cnt` field of the corresponding `SPT` entry against the `cntr` field above in the `DES` structure. If there is no change, then return `false`. But if there is a change, check the `op` condition match. If the `op` condition fails, update the `cntr` field and return `false`. But if it matches, compare the current GMT time with the time stamp in the `SPT` entry (plus the specified delay). If we have not yet reached the latter time, then return `false`; otherwise, update the `cntr` field and return `true`.

The Collection time stamp included in a `GETS32` reply message to a State-based request is to match the State time stamp plus the delay specified in the request.

When `STTE` processes the multicast message, the `SPT` must be searched for a match on the DI, the unique identifier for a State, to find the entry to update. Also, when initializing a State-based data request, the `ScanDES` function must search the `SPT` to match the `devX` field. To avoid such searches, for efficiency, we can use the hash-based Name Table that is part of each front end system. It was designed to faciliate searching for 6-character analog channel names, but its implementation is more general, allowing for more than one type of "name." For this case, the "name" is the 4-byte State device DI. The table allows for up to 8K names. A front end is typically configured for a maximum of 1K or 2K analog channels. A few hundred State device DI's will not add much impact. The Name Table is housed in dynamic memory and is initialized empty when the front end resets. The State "name" entries are added to the table as they are announced via multicast. If a State request comes in before such an announcement, there is no problem, as a new entry is made as described above. No replies can occur before the announcement for that State arrives.